RenewLedger : Renewable energy management powered by Hyperledger Fabric

Nikita Karandikar
Electronic Engg. and Computer Science
University of Stavanger
Stavanger, Norway
nikita.r.karandikar@uis.no

Antorweep Chakravorty
Electronic Engg. and Computer Science
University of Stavanger
Stavanger, Norway
antorweep.chakravorty@uis.no

Chunming Rong
Electronic Engg. and Computer Science
University of Stavanger
Stavanger, Norway
chunming.rong@uis.no

***Abstract*—Trading and storage of renewable energy offers a way for the prosumer to extract value from the surplus energy that they produce, while also mitigating energy shortfall. Power companies can enlist prosumers in demand response strategies for grid stability and cost savings. We present RenewLedger, a blockchain-based framework for renewable energy transaction, storage management and direct-to-consumer demand response incentivization and gamification for peak shaving. We design and implement this system using Hyperledger Fabric and report on performance benchmarking experiments conducted using Hyperledger Caliper.**

***Index Terms*—blockchain, Hyperledger Fabric, Hyperledger Caliper, performance benchmarking, renewable energy, prosumer, EV, energy storage, demand response, gamification.**

I. Introduction

The benefits of integrating renewable energy into our energy infrastructure are well established. Governments and individuals are showing an increased interest in renewable energy, especially solar energy, as awareness rises [1]. Prosumers are a new type of energy consumer that generate some of the energy they use by means of renewable sources like solar energy. Solar energy, however, is uncertain and depends on the weather. At times, the prosumer may have surplus energy they cannot use, other times, the prosumer may have a shortfall. Trading excess energy with other prosumers or storing it can help ease some of this uncertainty. This requires the formulation of a solution to facilitate transactions between prosumers and to manage energy storage.

Blockchain [2] was born out of one of the most successful attempts at building a cryptocurrency. Bitcoin, [2] which was introduced in 2008, used an append-only decentralized shared ledger called block chain which was composed of blocks of transactions linked together in a chain. As this was a public system allowing anyone to join and perform transactions, many features were implemented to make malicious activity difficult. Bitcoin ensured ordering of blocks by cryptographically linking each block to its predecessor by storing a hash of the previous block in each block. Moreover, the ledger existed on many nodes simultaneously, so any node unilaterally changing its own copy would not be accepted in the network. In order to get the privilege of adding transactions, nodes race to perform a computationally intensive operation called Proof of Work which requires time and extensive resources to calculate but is trivial to verify. Nodes are incentivized to contribute resources to the network. The ledger developed by the creators of Bitcoin has received interest in industry and academia as practitioners find it fulfils other use cases [3].

There are two broad types of blockchain networks: permissioned and permissionless. Permissionless networks such as Bitcoin are open to all and anyone is permitted to propose transactions on the system. Permissioned networks are formed of authenticated nodes with clearly defined access privileges in the network. Organizations that transact with each other in a business network usually have their own disparate systems. While this centralized system gives them more control, it becomes a single point of failure. As the systems are not necessarily built to function seamlessly with each other, establishing provenance of an asset being traded becomes a laborious task. A permissoned blockchain network can provide a unified identity management and provenance tracing system. Consensus in a permissioned blockchain network is achieved based on the agreed upon endorsement policy, allowing organizations to simulate and endorse transactions concerning them before they are processed. This allows them to do away with energy intensive consensus mechanisms like Proof of Work and cryptocurrencies to incentivize nodes to contribute computational resources. Permissioned networks also help companies fulfil the Know your customer (KYC) and anti-money laundering (AML) obligations [4] imposed by various governments. Hyperledger Fabric [5] is one of the most widely used enterprise level permissioned blockchain platforms. It is an open source Linux Foundation project and has a development committee of over 200 developers and 35 organizations. It has a modular design allowing operators to tailor the implementation to the use case by supporting different implementations of consensus, membership management and transaction data format. Self enforcing smart contracts can be coded in popular languages such as Java, Go and Node.Js, thus eliminating the need for a developer to learn a new domain specific language. Moreover, developers are not

This research was funded by the Project no 267967: Energix of NFR (Norwegian Research Council)

978-1-7281-8086-1/20/$31.00 ©2020 IEEE

restricted to predefined tokens for transactions and an asset can be defined as anything of value.

In our previous work [6], we presented a theoretical basis for a blockchain based energy trading and storage management system. In this work, we implement such a system and conduct performance benchmarking experiments for various scenarios. We use Hyperledger Caliper [7], which is another project under the Hyperledger umbrella, to conduct our benchmarking experiments. Caliper uses the Common Connection Profile (CCP) of the Fabric Software Development kit (SDK) that allows it to implement complex scenarios and take advantage of many Fabric features.

The rest of the paper is organized as follows. Section II presents an overview of system, in Section III we discuss the implementation and in Section IV we present and analyse the results of our experiments. We present our work in the context of related works in Section V and we conclude in Section VI.

II. Overview of System

We consider the following capabilities of the system:

1. Transacting between Prosumers
2. Prosumer Incentivization
3. Prosumer gamification
4. Managing community level energy storage transactions
5. Managing use of Electric Vehicle (EV) battery as storage
6. Managing EV battery charging transactions

A. Application entities

1) Trader: The Trader is a user of the system who wishes to transact units and does so using the Trading Platform. The Trader can perform activities such as transacting surplus energy with other Traders and EVs, using storage facilities for excess energy by participating in a community level storage or storing energy in EV batteries rented out for this purpose. The Trader can also earn tokens by participating in demand response incentivization and gamification tasks offered by the Power Company.

2) Trading Platform: The Trading Platform lists available tokens for bidding on or transacting with. We consider bidding as expressing binding interest in a token without negotiating the price, but this can be tailored to the use case. The Trading Platform provides a client for Traders to interact with the network and facilitates transactions. The Trading Platform can include representatives of auditory bodies or government entities to monitor or administer the system.

3) Power Company: The Power Company can directly reach prosumers to aid its demand response strategies by incentivizing them for their participation. They can offer direct incentives to the prosumer for accomplishing tasks such offering up their projected consumption for the given time period, agreeing to being monitored by the Power Company for compliance and receiving reward tokens for tasks completed or having the agreed value taken away as penalty for reneging. Another way the Power Company can enlist the prosumer in their peak shaving efforts is to offer a list of games, which are tasks with associated rewards and penalties presented in a gamified context. Games can include tasks like not running the air conditioning on a particularly hot day or moving dishwasher or laundry machine use to off peak hours. Also, if the Power Company has access to the data about energy stored in the Community level storage, it could help inform their demand response strategy.

4) Electric Vehicle: The electric vehicles can buy surplus energy from the prosumers to charge their batteries. They can also earn reward tokens by renting out the EV battery to be used as storage devices when not in use.

5) Storage as a Service Provider: The Storage as a Service provider (hereafter Storage Provider) handles all the tasks involved with setting up and managing a community level storage for surplus energy and abstracts out the intricacies of these activities for the users.

B. Architecture

1) Organizations: Organizations in the Hyperledger Fabric architecture logically map the different organizations in our system. The Trading Platform, the Power Company and the Storage Provider are the three organizations in our architecture. Each organization has two peers for redundancy. The leader peer is the peer that receives blocks from the orderer and then transmits them to the other peers in the organization using gossip protocol. The leader peer in each organization is chosen through election and peers are load balanced.

2) Clients: We setup one client for each organization. The Trading Platform client is used by the Traders to transact energy units, participate in gamification and incentivization tasks and for storing surplus energy. The Power Company uses its client to post gamification and incentivization tasks and to process the rewards. The Storage Provider organization processes storage requests and rewards using its client.

3) Orderer Organization: The ordering service nodes in the Hyperledger Fabric network are responsible for putting all the transactions in the correct order and creating blocks of transactions for the peers to validate and commit. As this is an important function, the ordering service nodes are not under the control of any one organization but are members of an orderer organization. This is a separate organization administered by certificate authorities from all other organizations.

4) Certificate Authorities: A certificate authority (CA) is present in each organization. The CA issues identity certificates to member components which components use to identify each other. The identity determines a user's access within the channel.

5) Chaincodes: The blockchain developer encodes the business logic of the application into the chaincode. A chaincode can have many smart contracts that cover the governance rules for different interactions between the transacting parties. The chaincode must be installed and instantiated on the channel before it is called. When specifying a chaincode in the CCP, we specify target peers on which this chaincode will be installed. For instance, as the trading of energy units between prosumers should only involve the Traders and the Trading Platform, we specify the target peers of this chaincode

to be members of the Trading Platform organization. Another option is to create separate channels for each set of trading relationships, but this was not explored in this work.

III. Implementation

A. Tokens

We consider six different types of tokens in our application reflecting the six use cases mentioned in Section II. We see the relationship between the different application entities and the tokens in Figure 1. Each token has a key field and value field containing four values. The values depend on the functionality for which the token is used. Updating and Querying the world state are time consuming operations and so to improve performance, we have chosen LevelDB which is the more performant choice [8]. LevelDB is a key value storage that does not provide rich querying so we use the key of the asset in order to store important information.

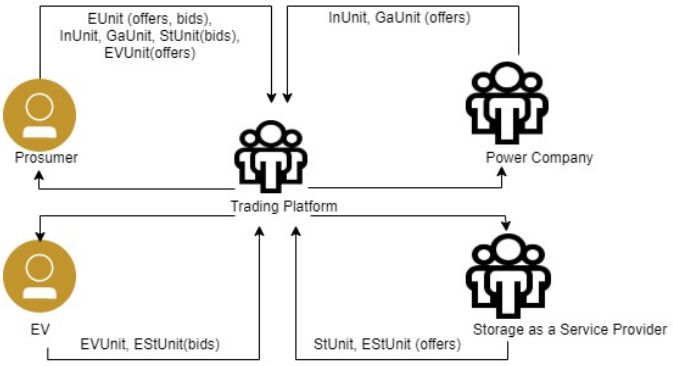

Fig. 1. Relationship between Application Components

1) Token key: The token key is composed of the serial number of the token and the token type and has the format :serial number-token type. There are 6 types of tokens.

2) Types of Tokens: The following are the types of tokens
1. EUnit, EVUnit:
We use the EUnit token to represent energy units traded between two prosumers and the EVUnit token to represent energy units sold by a prosumer to an EV. The values in these tokens:
a)price- the price of the token
b)location- location of the seller
c)value- amount of electricity in the token
d)bid-true/false showing whether a user has bid on this unit
2. InUnit, GaUnit:
The InUnit and GaUnit tokens are created by the Power Company for Incentivization and Gamification respectively and are offered to the user. The values in these token:
a)reward- incentive for completing the prescribed task
b)penalty- for agreeing and then failing to complete the task
c)requirement- what the task entails
d)bid- true/false has a user chosen this particular token
3. StUnit, EStUnit :
The StUnit tokens are used by the user to access the community level storage facility in order to store their surplus energy. The community level storage facility may also include unused EV batteries. The values in these token:
a)price- to store a given amount of energy
b)value- amount of energy storage offered with this token
c)conditions - of storage
d)bid- true/false has a user chosen this particular token
The EStUnit tokens are reward tokens generated by the Storage Provider in order to supplement its storage capacity with EV batteries. The values in these token:
a)reward -associated with participating
b)amount- of energy that will be stored, or capacity required
c)conditions - of storage
d)bid- true/false has a user chosen this particular token

IV. Experiments

A. Setup

Our experiments were conducted using Hyperledger Fabric V1.4.6 to build the system under test and Hyperledger Caliper V0.2 was the performance benchmarking tool. We used Ubuntu 16.04 on a machine with an Intel Core i7 processor, 4 cores CPU, 32 GB RAM, 256 GB SSD.

1) Experimental Parameters: Endorsement Policy: 1 member of each organization to endorse each transaction.
Consensus: RAFT with 3 Ordering Service Nodes [9]
We conducted two types of operation for each network configuration: read and write. We chose send rates in Transactions per second (TPS) based upon our infrastructure capabilities:
a) Read Operation:
Total transactions performed for each data point: 1000
Send Rate (in TPS): 25, 50, 100, 200, 400
b) Write Operation:
Total transactions performed for each data point: 500
Send Rate (in TPS): 5, 10, 20, 40, 80

B. Transaction flow and Performance

1) Tokens involving 1 Organization: Figure 2, shows the transaction flow for EUnit and EVUnit. Transacting electricity units between a buyer and a seller requires both users to be registered on the Trading Platform before submitting a transaction proposal. This transaction would involve a read from the world state by the buyer to see what Units are available for purchase and the associated price. An update transaction would then be initiated to bid on the Unit. The seller would read the state and initiate sale. This would involve two updates to the world state: the ownership of the token will be transferred from the seller to the buyer and a payment token to be transferred from the buyer to the seller. The only organization involved in this transaction is the Trading Platform. The user will initiate a transaction, whether read or write/update using a Client that uses an Application SDK to construct a well formed transaction proposal and signs it by generating a unique signature using its credentials stored in its wallet. This proposal is then sent to the appropriate number of endorsement peers as specified in the endorsement policy.

In this case it only goes to the Trading Platform organization, which verifies that the proposal is well-formed and signed correctly by a user with the access to request this operation. It executes it against the world state and sends the generated read/write set back to the client with an endorsement. If the operation is a Read, then the transaction flow stops here. If the operation requested is a write/update, the client inspects the response to see if the endorsement policy was met and broadcasts it to the orderer which orders the transactions, creates blocks of transactions and sends them to the peer organizations. The peers verify the endorsements, that the read/write set has not been changed since creation and then update their world state and commit the block to their ledger.

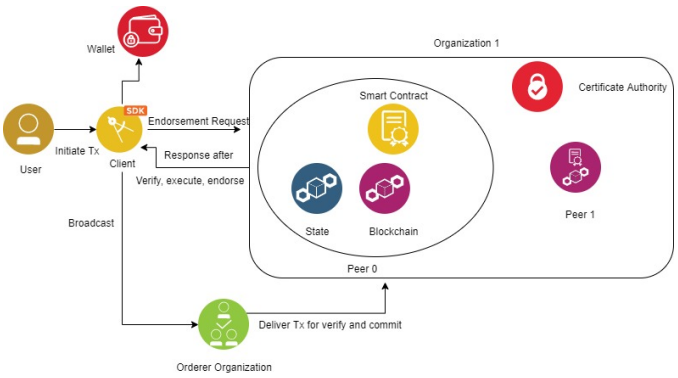

Fig. 2. Transaction Flow involving 1 Organization: EUnit, EVUnit

Figure 3, shows the performance of a read operation involving a single organization. The throughput increases with increase in send rate and even at 400 TPS send rate, the throughput is close to the send rate. This shows that the system has the potential to process even higher send rates. The read latency stays well below 1 second for all data points shown and interestingly, it reduces slightly for higher send rates. The Hyperledger Caliper tool automatically load balances between the available peers. The endorsement at the peers can run concurrently, allowing different peers to handle different transactions.

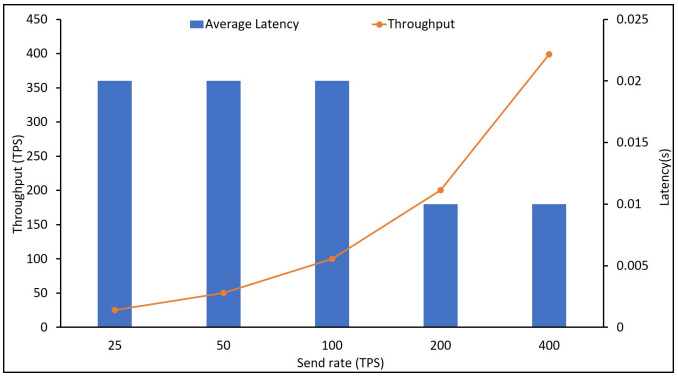

Fig. 3. Read Operation involving 1 Organization: EUnit, EVUnit

Figure 4, shows the performance profile for a write operation involving a single organization. The throughput of the write operation is lower that that of the read operation and the latency is higher. The write operation involves more steps as explained earlier in this subsection. The ordering operations and ledger update are time consuming operations leading to a higher latency for write operation as compared to read.

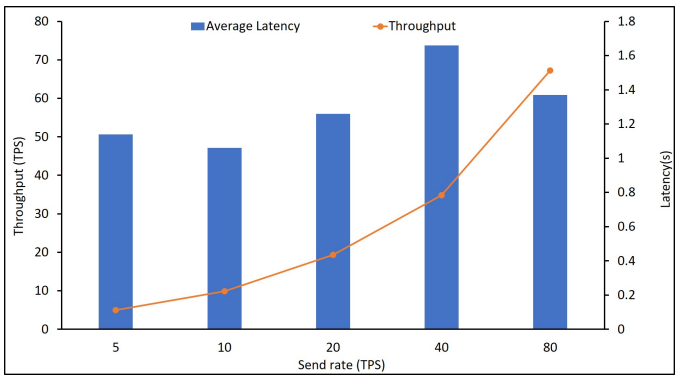

Fig. 4. Write Operation involving 1 Organization: EUnit, EVUnit

2) Tokens involving 2 Organization: Figure 5, shows the transaction flow for InUnit and GaUnit. In this scenario, the Trading Platform and the Power Company are the two Organizations involved and one peer from each organization must approve each transaction. The client, must thus be authenticated by the Certificate Authorities of both organizations and both organizations will receive an endorsement request from the client. A user from the Power Company would access the network using the Client associated with that organization and perform write transactions, adding in different InUnit and GaUnit tokens as needed by the organization for its demand response strategy. The Trader or user will interact with the system using the Trading system Client and perform read requests to see what tokens are available and the associated rewards for playing correctly, penalties for reneging and conditions. The user will then perform write/update requests bidding upon the InUnits and GaUnits, they are interested in. The Power Company reads these bids and monitors compliance and then initiates a change of ownership write request for the tokens. The business logic to ascertain whether a change of ownership will happen and the particulars of that transaction would be encoded in the chaincode and would depend on the specific implementation and their business needs and would impact performance based on their complexity. We focus on the actual interactions whether read or write with the blockchain network that these transactions would boil down to once these determinations are made.

Figure 6 shows the performance of Read operation involving two organizations. The throughput is close to the send rate in this case and the latency is very low, under 1 second in each case studied, showing that the system is not yet saturated. However, it does have a higher latency and slightly lower throughput as compared to the corresponding one organization values. As the number of Organizations increase, the number of endorsements required for each transaction increase, impacting the performance.

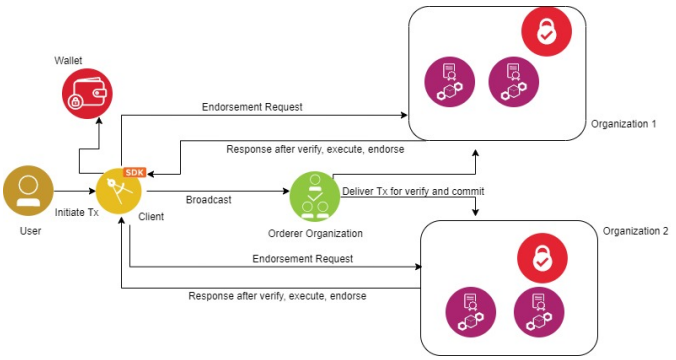

Fig. 5. Transaction flow involving 2 Organizations: InUnit, GaUnit

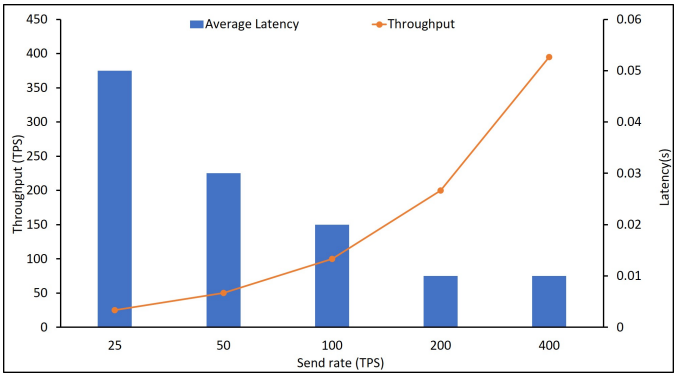

Fig. 6. Read Operation involving 2 Organizations: InUnit, GaUnit

Figure 7, shows the performance of the Write operation involving two organizations. The latency increases sharply at 40 TPS send rate from 1.44 s to 3.03 s. The throughput still increases with the increase in send rate. In this case, the number of endorsements required and also the number of peers that must validate and commit each transaction increases compared to the Write operation involving 1 Organization. Thus, this case has a higher latency and lower throughput as compared to corresponding the one organization write operations.

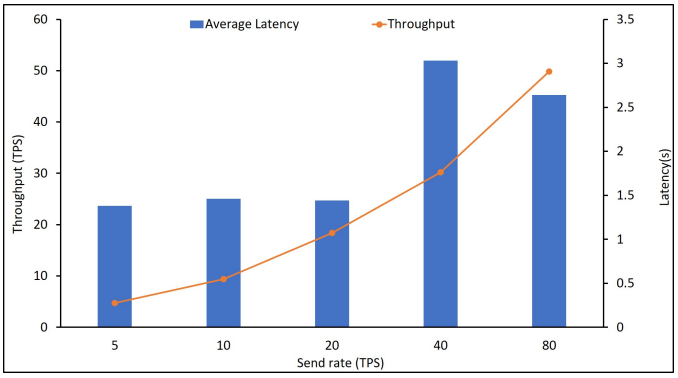

Fig. 7. Write Operation involving 2 Organizations: InUnit, GaUnit

3) Tokens involving 3 Organization: Figure 8, shows the transaction flow for StUnit and ESTUnit tokens. Three Organizations are involved in this scenario- Trading Platform, Storage Provider and Power Company. The Storage Provider would subsume the EV battery in its community level storage facility and abstract this detail from the prosumer. The Prosumer is looking to store surplus units of electricity and conducts a read operation to check how many StUnits they have which can be used to rent storage capacity with the Storage as a Service Provider. The prosumer would then initiate a write operation to transfer the ownership of the StUnit to the Storage Provider. The Power Company could be an active member of this scenario by providing endorsing peers or it could just have its peers be committing peers. The Power Company would then have data on not only the stored energy at all times (world state) but also the data for past storage (ledger data) and could factor it into their demand response calculations. This would be dictated by the exact business use case and relationship. In our experiments, we consider that all three organizations have endorsing peers and that the endorsement policy requires one peer of each organization to endorse each transaction.

The EV owner would earn ESTUnits from the Storage Provider for their participation in the community level storage facility. This transaction would start when the EV owner submits a write operation by bidding on one of the offered ESTUnits. The Trading Platform, the Storage Provider and if applicable, the Power Company must endorse this. When the task is successfully completed, all three organizations endorse a proposal to transfer ownership of the earned ESTUnit to the EV Owner.

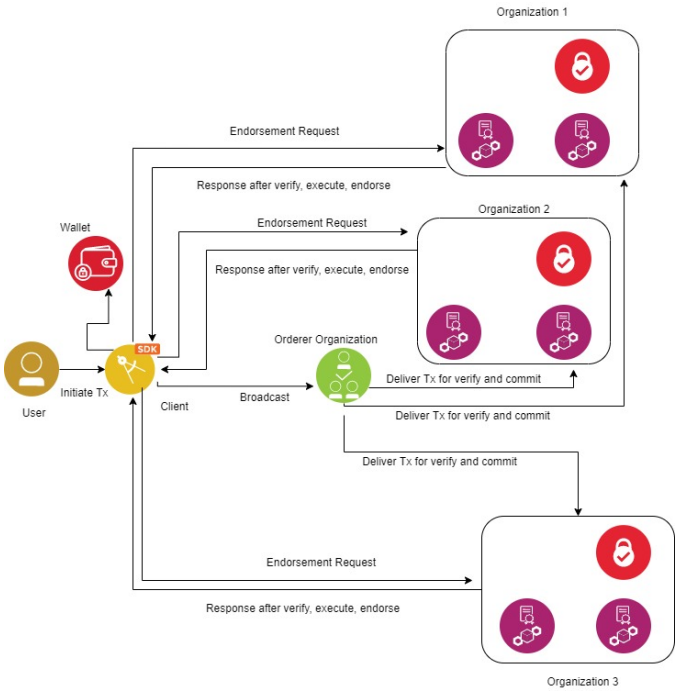

Fig. 8. Transaction Flow involving 3 Organizations: StUnit,ESTUnit

Figure 9 shows the performance of Read Operation for a transaction involving three Organizations. Compared to the read operation for one and two organizations, the latency is

higher and throughput is lower in each case. At 400 TPS send rate, the latency starts to rise again due to the bottleneck of the number of available endorsement peers. The latency while higher for this configuration was still well below 1 second and the effect of change in send rate was miniscule.

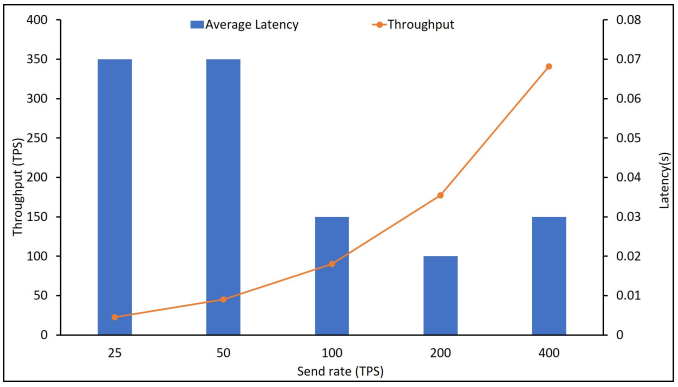

Fig. 9. Read Operation involving 3 Organizations: StUnit, ESTUnit

Figure 10 shows the performance of a Write Operation involving three organizations. The latency increases rapidly from 2.01 seconds at 20 TPS send rate to 4.37 seconds at 40 TPS send rate and the throughput while still increasing begins to level off. The write operation for three organizations had the highest latency and lowest throughput of all the configurations presented.

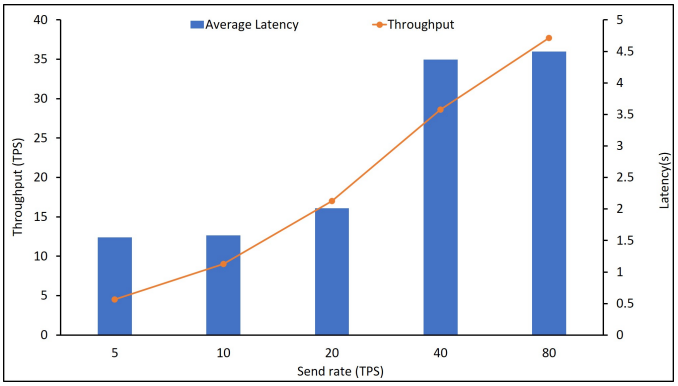

Fig. 10. Write Operation involving 3 Organizations: StUnit, ESTUnit

V. Related Works

Pipattanasomporn et al [10] and Jogunola et al [11] have published their respective articles discussing energy trading on the blockchain. Their works target different use cases from ours and their implementations were built using Hyperledger Composer which has now been deprecated. Hyperledger Fabric provides a much more customizable platform for building blockchain solutions than Composer. Long et al [12] investigated three market paradigms to reduce costs for customers trading energy in a community microgrid. Park et al [13] developed their solution on the IBM Blockchain Platform which is a proprietary platform and is not free or open source. Also, they have targetted different use cases than we have. Saxena et al [14] explore the impact of various bidding strategies used by energy consumers and present a market price clearing algorithm. Our work presents a framework for energy management implemented using Hyperledger Fabric.

VI. Conclusion

In this paper, we presented RenewLedger, a Hyperledger Fabric based framework for renewable energy management. We extended our previous work 'Transactive energy on Hyperledger Fabric' and explained the design and implementation of this system in detail. We also conducted benchmarking experiments to evaluate the performance of the system with read and write operation for transactions involving one, two and three organizations. We found that, for our implementation, read operations were executed with latency under 1 second and throughput close to send rate for all three configurations for send rates up to 400 TPS. The write operations showed significant difference in latency based on the configuration and went from under 2 sec for 1 Organization to up to 4.5 for three organizations. The throughput also decreased from almost 70 TPS in 1 Organization for a 80 TPS send rate to under 40 TPS for 3 Organizations for the same send rate.

References

[1] E. P. I. Association *et al.*, "Global market outlook for photovoltaics 2014-2018," *EPIA, Brussels*, 2014.

[2] S. Nakamoto *et al.*, "Bitcoin: A peer-to-peer electronic cash system," 2008.

[3] M. Crosby, P. Pattanayak, S. Verma, V. Kalyanaraman, *et al.*, "Blockchain technology: Beyond bitcoin," *Applied Innovation*, vol. 2, no. 6-10, p. 71, 2016.

[4] P. Makin and C. Hyperion, "Regulatory issues around mobile banking," *The Development Dimension ICTs for Development Improving Policy Coherence: Improving Policy Coherence*, vol. 139, 2010.

[5] "Hyperledger fabric documentation release 1.4," *Hyperledger*, 2019.

[6] N. Karandikar, A. Chakravorty, and C. Rong, "Transactive energy on hyperledger fabric," in *2019 Sixth International Conference on Internet of Things: Systems, Management and Security (IOTSMS)*, pp. 539–546, IEEE, 2019.

[7] "Hyperledger caliper documentation," 2020.

[8] P. Thakkar *et al.*, "Performance benchmarking and optimizing hyperledger fabric blockchain platform)," *2018 IEEE 26th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS).*, pp. 264–276, 2018.

[9] D. Ongaro and J. Ousterhout, "In search of an understandable consensus algorithm.," *2014 USENIX Annual Technical Conference (USENIXATC 14)*, pp. 305–319, 2014.

[10] M. Pipattanasomporn, M. Kuzlu, and S. Rahman, "A blockchain-based platform for exchange of solar energy: Laboratory-scale implementation," in *2018 International Conference and Utility Exhibition on Green Energy for Sustainable Development (ICUE)*, pp. 1–9, IEEE, 2018.

[11] O. Jogunola, M. Hammoudeh, B. Adebisi, and K. Anoh, "Demonstrating blockchain-enabled peer-to-peer energy trading and sharing," in *2019 IEEE Canadian Conference of Electrical and Computer Engineering (CCECE)*, pp. 1–4, IEEE, 2019.

[12] C. Long, J. Wu, C. Zhang, L. Thomas, M. Cheng, and N. Jenkins, "Peer-to-peer energy trading in a community microgrid," in *2017 IEEE power & energy society general meeting*, pp. 1–5, IEEE, 2017.

[13] I. H. Park, S. J. Moon, B. S. Lee, and J. W. Jang, "A p2p surplus energy trade among neighbors based on hyperledger fabric blockchain," in *Information Science and Applications*, pp. 65–72, Springer, 2020.

[14] S. Saxena, H. Farag, A. Brookson, H. Turesson, and H. M. Kim, "Design and field implementation of blockchain based renewable energy trading in residential communities," *arXiv preprint arXiv:1907.12370*, 2019.